3.8）死信队列(死信交换机)

一:死信队列DLX(Dead-leater-exchange)

1.1）**什么是死信**?

就是在队列中的消息如果没有消费者消费，那么该消息就成为一个死信，那这个消息被重新发送到另外一个exchange**上的话，**

那么后面这个exhcange**就是死信队列**

1.2)**消息变成死信的几种情况**

消息被拒绝：（basic.reject/basic.nack**）并且**requeue(**重回队列**)**的属性设置为** false **表示不需要重回队列，那么该消息就是一个死信消息**

消息TTL**过期**

消息本身设置了过期时间，或者队列设置了消息过期时间x-message-ttl

队列达到最大长度:**比如队列最大长度是**3000 **,那么**3001**消息就会被送到死信队列上.**

1.3)**死信队列也是一个正常的**exchange,**也会通过**routingkey **绑定到具体的队列上。**

1.4)**代码演示;**

```
public class DLX_CustomConsumer extends DefaultConsumer {
private Channel channel;
/**
 * Constructs a new instance and records its association to the passed-in channel.
 *
 * @param channel the channel to which this consumer is attached
 */
 public DLX_CustomConsumer(Channel channel) {
 super(channel);
 this.channel = channel;
 }
public void handleDelivery(String consumerTag,
 Envelope envelope,
 AMQP.BasicProperties properties,
 byte[] body)
 throws IOException
 {
 System.out.println("为了测试死信队列,我们进行nack");
 //把消息变为死信 通过nack 且requeue不进行重新发送
 channel.basicNack(envelope.getDeliveryTag(),false,false);
 }
}
```

==========消费者

```
package com.hnnd.mq.dlx;

import com.rabbitmq.client.Channel;
import com.rabbitmq.client.Connection;
import com.rabbitmq.client.ConnectionFactory;

import java.io.IOException;
import java.util.HashMap;
import java.util.Map;
import java.util.concurrent.TimeoutException;

/**
 * Created by Administrator on 2018/10/19.
 */
public class DLX_consumer {

    public static void main(String[] args) throws IOException, TimeoutException {
        ConnectionFactory connectionFactory = new ConnectionFactory();
        connectionFactory.setVirtualHost("/");
        connectionFactory.setHost("47.104.128.12");
        connectionFactory.setPort(5672);

        Connection connection = connectionFactory.newConnection();

        Channel channel = connection.createChannel();

        String normalExchangeName= "test.normaldlx.exchange";
        String normalQueueName = "test.normaldlx.queue";

        String dlxExchangeName="test.dlx.exchange";
        String dlxQueueName = "test.dlx.queue";

        //声明一个正常的业务队列
        channel.exchangeDeclare(normalExchangeName,"topic",true,true,false,null);
        Map<String,Object>  argurments= new HashMap<>() ;
        //设置正常队列中的死信发往哪个队列
        argurments.put("x-dead-letter-exchange", "test.dlx.exchange");
        channel.queueDeclare(normalQueueName,true,true,true,argurments);
        channel.queueBind(normalQueueName,normalExchangeName,"test.normaldlx.key");

        //声明死信队列
        channel.exchangeDeclare(dlxExchangeName,"topic",true,true,false,null);
        channel.queueDeclare(dlxQueueName,true,true,true,null);
        channel.queueBind(dlxQueueName,dlxExchangeName,"#");

        channel.basicConsume(normalQueueName,false,new DLX_CustomConsumer(channel));

    }
}
```

```
==========================================生产者
package com.hnnd.mq.dlx;

import com.rabbitmq.client.AMQP;
import com.rabbitmq.client.Channel;
import com.rabbitmq.client.Connection;
import com.rabbitmq.client.ConnectionFactory;

import java.io.IOException;
import java.util.concurrent.TimeoutException;

/**
 * Created by Administrator on 2018/10/19.
 */
public class DLX_Producter {

    public static void main(String[] args) throws IOException, TimeoutException {
        ConnectionFactory connectionFactory = new ConnectionFactory();
        connectionFactory.setVirtualHost("/");
        connectionFactory.setHost("47.104.128.12");
        connectionFactory.setPort(5672);

        Connection connection = connectionFactory.newConnection();

        Channel channel = connection.createChannel();

        //设置消息5S钟超时
        AMQP.BasicProperties basicProperties = new AMQP.BasicProperties().builder()
                .expiration("10000").build();

        channel.basicPublish("test.normaldlx.exchange","test.normaldlx.key",basicProperties,"测试消息转为死信队列".getB
    }
}
```